Wybrane elementy praktyki projektowania oprogramowania

Wykład 12/15
Relacyjne bazy danych (1)

Wiktor Zychla 2018/2019

1 Spis treści

2 Klasyfikacja baz danych

2.1 Hierarchiczne bazy danych

Pomysł na organizację danych w strukturę hierarchiczną jest dość stary i w taki sposób początkowo implementowano bazy danych. Współcześnie podejście hierarchiczne wykorzystują m.in. pliki XML.

Zalety:

- Proste i szybkie operacje z uwagi na prostą, drzewiastą implementację

Wady:

- Ograniczenie relacji z węzła do tylko jednego węzła-rodzica
- Zmiana modelu danych może skutkować koniecznością przeprojektowania struktury

2.2 Relacyjne bazy danych

W tej implementacji uogólnia się pojęcie relacji pozwalając elementowi być jednoznacznie identyfikowanym i być w relacji z innym elementem z tej samej lub innej struktury (nazwanej tu: tabelą).

Obsługa relacyjnej bazy danych jest ustandaryzowana – silnik bazy danych obsługuje język Structured Query Language (SQL). Ponieważ język SQL został ustandaryzowany, obsługa różnych systemów baz danych z punktu widzenia programowania zapytań do danych wygląda identycznie.

Tabela 1 Rewizje standardu języka SQL, za https://en.wikipedia.org/wiki/SQL

Year	Name	Alias	Comments
1986	SQL-86	SQL-87	First formalized by ANSI.
1989	SQL-89	FIPS127-1	Minor revision that added integrity constraints, adopted as FIPS 127-1.
1992	SQL-92	SQL2, FIPS 127-2	Major revision (ISO 9075), *Entry Level* SQL-92 adopted as FIPS 127-2.
1999	SQL:1999	SQL3	Added regular expression matching, recursive queries (e.g. transitive closure), triggers, support for procedural and control-of-flow statements, non-scalar types (arrays), and some object-oriented features (e.g. structured types). Support

			for embedding SQL in Java (SQL/OLB) and vice versa (SQL/JRT).
2003	SQL:2003		Introduced XML-related features (SQL/XML), *window functions*, standardized sequences, and columns with auto-generated values (including identity-columns).
2006	SQL:2006		ISO/IEC 9075-14:2006 defines ways that SQL can be used with XML. It defines ways of importing and storing XML data in an SQL database, manipulating it within the database, and publishing both XML and conventional SQL-data in XML form. In addition, it lets applications integrate queries into their SQL code with XQuery, the XML Query Language published by the World Wide Web Consortium (W3C), to concurrently access ordinary SQL-data and XML documents.[34]
2008	SQL:2008		Legalizes ORDER BY outside cursor definitions. Adds INSTEAD OF triggers, TRUNCATE statement,[35] FETCH clause.
2011	SQL:2011		Adds temporal data (PERIOD FOR)[36] (more information at: Temporal database#History). Enhancements for *window functions* and FETCH clause.[37]
2016	SQL:2016		Adds row pattern matching, polymorphic table functions, JSON.

Przykłady baz relacyjnych to:

- Oracle Database
- Microsoft SQL Server
- MySQL
- PostgreSQL

2.3 Obiektowe bazy danych

Jeśli zamiast tabel i wierszy, w bazie danych zostaną zapisane bezpośrednio obiekty (te zainicjowane z poziomu języka programowania, np. C++) a zamiast relacji między wierszami tabel w bazie zostaną również zapamiętane wskaźniki między obiektami – to mówimy o bazie obiektowej.

Zalety:

- Skoro kod aplikacji powstaje i tak w języku obiektowym, to składanie do bazy bezpośrednio obiektów oznacza, że odpada konieczność zamiany obiektów na wiersze tabeli (tzw. object-relational impendance mismatch)

Wady:

- Baza obiektowa bywa „uwiązana" do konkretnego języka programowania
- Jest zwykle zauważalnie mniej wydajna od bazy relacyjnej
- Istnieje język zapytań – Object Query Language (OQL) ale jego implementacje są rzadkie (istnieją praktyczne systemy baz obiektowych jak db4o które mają problemy ze wsparciem OQL). Język OQL nie ma takiej podbudowy w modelu algebraicznym jak SQL, stąd problemy z samą semantyką języka.
- zamiast zapytań OQL używa się innych technik zapytań, np. obiekt wzorcowy i prosi bazę danych o obiekty „podobne" do wzorcowego (tzw. Query By Example)
- często brak wsparcia dla indeksów

2.4 Bazy grafowe

Baza grafowa to specjalizowana implementacja bazy zoptymalizowanej pod kątem relacji między elementami. W implementacji zapytań wykorzystuje się dobrze znane algorytmy grafowe, stąd – istnieją pewne klasy problemów w których baza grafowa działa znacznie szybciej niż relacyjna. Baza grafowa ma swój własny jezyk zapytań i – najczęściej – każda własny, ponieważ nie ma de facto jednego standardu zapytań grafowych.

Przykładem bazy grafowej jest neo4j.

2.5 Bazy NoSQL

Szeroka kategoria baz tzw. NoSQL obejmuje wiele specjalizowanych implementacji, dedykowanych zadaniom które przez specyficzną implementację mogą być realizowane szybciej niż w bazie relacyjnej

(stąd – bazy grafowe są również uznawane za przedstawiciela rodziny „NoSQL")

Przewagi implementacyjne mogą obejmować m.in.:

- Łatwiejszą skalowalność
- Lepszą optymalizację wyszukiwania tekstowego

Wady to m.in.:

- Brak jednolitego standardu zapytań
- Trudność w odwzorowywaniu złożonych modeli danych
- Trudność „wersjonowania" modelu

Przykłady baz NoSQL

- MongoDB
- CouchDB
- MemcachedDB

3 Podstawy baz relacyjnych

W trakcie wykładu zobaczymy praktyczną demonstrację przykładowego systemu relacyjnych baz danych. Opowiemy o sposobach obsługi bazy danych.

Omówimy poszczególne elementy składowe bazy danych:

- Tworzenie, kopia zapasowa
- Schematy, tabele
- Kolumny – typy, wartość NULL
- Klucze główne (primary key)
- Klucze obce (foreign key)
- Modelowanie relacji
    - jeden-jeden
    - jeden-wiele
    - wiele-wiele
    - self-join

Omówimy również podstawowe typy kwerend

- kwerendy SELECT, INSERT, UPDATE, DELETE
- klauzule WHERE, ORDER BY, GROUP BY, JOIN
- funkcje agregujące COUNT, AVG, MIN, MAX

Skrypt bazy którą tworzyliśmy na wykładzie

```
CREATE TABLE [dbo].[Child](
 [ID] [int] IDENTITY(1,1) NOT NULL,
 [ChildName] [nvarchar](150) NOT NULL,
 [ID_PARENT] [int] NOT NULL,
 CONSTRAINT [PK_Child] PRIMARY KEY CLUSTERED
(
 [ID] ASC
)
) ON [PRIMARY]
GO


CREATE TABLE [dbo].[Parent](
 [ID] [int] IDENTITY(1,1) NOT NULL,
 [ParentName] [nvarchar](150) NOT NULL,
 CONSTRAINT [PK_Parent] PRIMARY KEY CLUSTERED
(
 [ID] ASC
)
) ON [PRIMARY]

ALTER TABLE [dbo].[Child]  WITH CHECK ADD  CONSTRAINT
[FK_Child_Parent] FOREIGN KEY([ID_PARENT])
REFERENCES [dbo].[Parent] ([ID])
```